sklearn.preprocessing.LabelEncoder

» *class* sklearn.preprocessing.**LabelEncoder** [source]

Encode labels with value between 0 and n_classes-1.

Read more in the User Guide.

Attributes: **classes_** : array of shape (n_class,)

Holds the label for each class.

Examples

LabelEncoder can be used to normalize labels.

```
>>> from sklearn import preprocessing
>>> le = preprocessing.LabelEncoder()
>>> le.fit([1, 2, 2, 6])
LabelEncoder()
>>> le.classes_
array([1, 2, 6])
>>> le.transform([1, 1, 2, 6])
array([0, 0, 1, 2]...)
>>> le.inverse_transform([0, 0, 1, 2])
array([1, 1, 2, 6])
```

It can also be used to transform non-numerical labels (as long as they are hashable and comparable) to numerical labels.

```
>>> le = preprocessing.LabelEncoder()
>>> le.fit(["paris", "paris", "tokyo", "amsterdam"])
LabelEncoder()
>>> list(le.classes_)
['amsterdam', 'paris', 'tokyo']
>>> le.transform(["tokyo", "tokyo", "paris"])
array([2, 2, 1]...)
>>> list(le.inverse_transform([2, 2, 1]))
['tokyo', 'tokyo', 'paris']
```

Methods

fit(y)	Fit label encoder
fit_transform(y)	Fit label encoder and return encoded labels
get_params([deep])	Get parameters for this estimator.
inverse_transform(y)	Transform labels back to original encoding.
set_params(**params)	Set the parameters of this estimator.
transform(y)	Transform labels to normalized encoding.

__init__()

Initialize self. See help(type(self)) for accurate signature.

fit(*y*) [source]

Fit label encoder

Parameters: **y** : array-like of shape (n_samples,)

Target values.

Returns: **self** : returns an instance of self.

»

fit_transform(*y*) [source]

Fit label encoder and return encoded labels

Parameters: **y** : array-like of shape [n_samples]

Target values.

Returns: **y** : array-like of shape [n_samples]

get_params(*deep=True*) [source]

Get parameters for this estimator.

Parameters: **deep: boolean, optional** :

If True, will return the parameters for this estimator and contained subobjects that are estimators.

Returns: **params** : mapping of string to any

Parameter names mapped to their values.

inverse_transform(*y*) [source]

Transform labels back to original encoding.

Parameters: **y** : numpy array of shape [n_samples]

Target values.

Returns: **y** : numpy array of shape [n_samples]

set_params(***params*) [source]

Set the parameters of this estimator.

The method works on simple estimators as well as on nested objects (such as pipelines). The former have parameters of the form `<component>__<parameter>` so that it's possible to update each component of a nested object.

Returns: **self** :

transform(*y*) [source]

»

Transform labels to normalized encoding.

Parameters: **y** : array-like of shape [n_samples]

Target values.

Returns: **y** : array-like of shape [n_samples]